Trabalho Pratico

Problema do Caixeiro Viajante (PCV)

AEDS II

Definição

O Problema do Caixeiro Viajante (do inglês *Travelling Salesman Problem* -TSP[1]) tem como objetivo determinar a menor rota para percorrer um conjunto de cidades, retornando sempre à cidade de origem. Esta rota é denominada ***ciclo*** hamiltoniano.

Por exemplo, dado o conjunto de cidades: São Paulo, Rio de Janeiro, Belo Horizonte e Brasília, e as distâncias entre elas (ver Tabela 1[2]), encontrar o menor ciclo entre estas cidades.

Cidades	São Paulo	Rio de Janeiro	Belo Horizonte	Brasília
São Paulo	0	436	587	1022
Rio de Janeiro	436	0	442	1166
Belo Horizonte	587	442	0	735
Brasília	1022	1166	735	0

Tabela 1. Distâncias em quilômetros entre as cidades.

O menor ciclo é:

Brasília – Belo Horizonte – Rio de Janeiro - São Paulo – Brasília

Total de quilômetros percorridos: **2630**

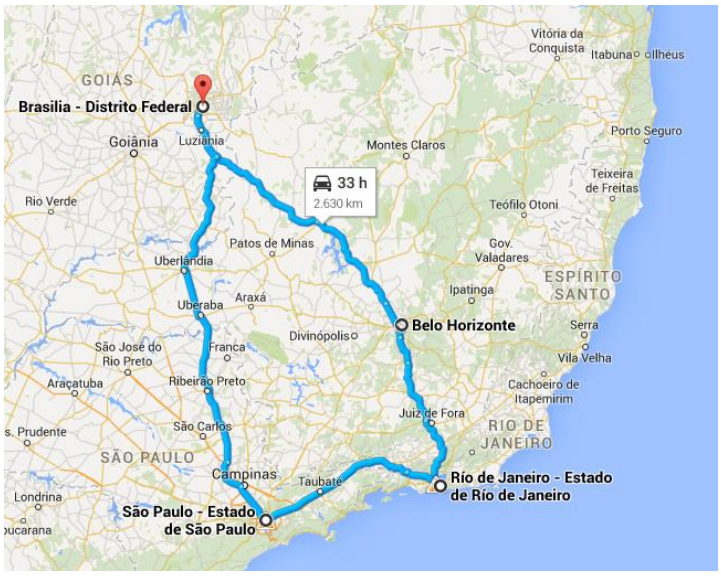

Figura 1. Mapa do menor percorrido entre 4 cidades brasileiras.[3]

O **Problema do Caixeiro Viajante Euclidiano** é um caso particular do **PCV** onde as distancias entre as cidades estão dadas por pontos num plano e obedecem a desigualdade triangular.

A distância euclidiana entre dois pontos A e B é dada por:

---

[1] http://en.wikipedia.org/wiki/Travelling_salesman_problem

[2] Dados obtidos do site: http://www.entrecidadesdistancia.com.br/ no 24/03/2015

[3] Imagem obtida do google maps (https://goo.gl/maps/Bzz4y) no 24/03/2015

$$d_{AB} = \sqrt{(a_x - b_y)^2 + (a_y - b_y)^2}$$

Quais quer três pontos A, B e C num espaço euclidiano satisfazem a *desigualdade triangular* dada por:

$$d_{AB} \leq d_{AC} + d_{CB}$$

Por exemplo, substituindo as distâncias do exemplo 1 por coordenadas euclidianas temos:

Cidades	Coordenadas	
	X	Y
São Paulo (SP)	2	0
Rio de Janeiro (RJ)	5	1
Belo Horizonte (BH)	4	4
Brasília (BSB)	1	10

Tabela 2. Coordenadas euclidianas das cidades.

A soma total das distâncias para o tour (ciclo) ótimo do exemplo anterior (BSB – BH – RJ – SP - BSB) é:

$$d_{T_1} = d_{BSB-BH} + d_{BH-RJ+} d_{RJ-SP+} d_{SP-BSB} \approx 22{,}08$$

[5pts] Parte I

Escreva uma função que recebe um vetor com as coordenadas dos pontos de um conjunto de cidades e imprime todos os possíveis ciclos de PCV Euclidiano. Você não deve repetir ciclos idênticos, mas que comecem por cidades diferentes. Por exemplo, o ciclo do exemplo 2, BSB – BH – RJ – SP – BSB é igual ao ciclo BH – RJ – SP – BSB – BH. Calcule a complexidade assintótica no pior caso de sua implementação.

Arquivo de entrada

A primeira fileira do arquivo de entrada tem o número de cidades (n). A primeira coluna contém um número que identifica a cidade. Na segunda coluna contém a coordenada X das cidades e na terceira coluna a coordenada Y.

n

1 x_1 y_1

2 x_2 y_2

3 x_3 y_3

4 x_4 y_4

...

Arquivo de saída

A primeira fileira do arquivo de saída tem o número ciclos gerados (m). Cada fileira vai conter um possível ciclo do PCV euclidiano, seguido do custo deste ciclo. As cidades que formam parte do ciclo devem estar separadas por espaço.

```
m
1     2     3     4 ... 1 = $custo
2     1     3     4 ... 2 = $custo
4     2     1     3 ... 4 = $custo
...
```

[2pts] Parte II

Dado o conjunto de *instâncias* (casos de teste) em anexo, com número de cidades igual a 6, 7, 8, 9, 10, 11, 12, 13, 14, ... Mostre **uma** solução ótima (aquela com menor custo) para cada instância e plote um gráfico do tempo de execução do seu algoritmo em função do número de cidades da instância.

O aluno que resolver a maior dentre as instâncias de teste, ganha 2 pontos extras!!!

[3pts] Parte III

Você deve solucionar o **PCV** usando a heurística do vizinho mais próximo (do inglês, *Nearest Neighbor*[4]), que consiste em iniciar o ciclo em uma cidade e continuar com a cidade mais próxima ainda não visitada

Você também deve calcular a complexidade assintótica no pior caso desta heurística e fazer uma tabela de comparação entre a solução da heurística e a solução ótima com as instâncias da **parte II**. A comparação é feita da calculando-se o *gap* de otimalidade (G) entre a solução ótima S* e a solução da heurística S:

$$G = \frac{S - S^*}{S^*}$$

[4] http://en.wikipedia.org/wiki/Nearest_neighbour_algorithm

Por exemplo, se o ciclo ótimo tem custo 22,08 e o ciclo que a heurística retorna tem custo 23,03. O gap da heurística nesta instância é:

$$G = \frac{23{,}03 - 22{,}08}{22{,}08} \approx 0{,}04 = 4\%.$$

Entrega

A entrega se divide em duas partes, que devem ser submetidas separadamente no Moodle:

(i) O aluno deve entregar o código, com um arquivo *Makefile* ou *Execute,* contendo todos os comandos que compilam o código. A linguagem para realizar este Trabalho é C ou C++. Se quiserem usar outra linguagem devem consultar previamente o professor. O código que deve ser entregado inclui três implementações diferentes:

- Parte I: Solução para o PCV com todos os ciclos
- Parte II: Solução Ótima (o ciclo com menor custo)
- Parte III: Solução usando Heurística do Vizinho mais Próximo

Cada uma destas partes deve gerar um arquivo de saída conforme foi anteriormente. Podem criar um executável para cada parte:

```
./tp1 <arquivo de entrada> <arquivo de saída>
```

Ou criar um executável só para todas as partes, neste caso seria:

```
./tp1 <parte> <arquivo de entrada> <arquivo de saída>
```

Exemplo executando o algoritmo pedido na parte I:

```
./tp1 1 cities12.txt solution_cities12.txt
```

(ii) O relatório deve estar em formato PDF e dividido em três sessões, cada uma delas contendo o que foi pedido em cada uma das três partes do TP.